WEBPACK 项目顺序

1. 首先初识化项目： cnpm init -y
2. 创建2个文件夹:
3. dist 存放打包后的 文件
4. src 存放原始文件

最简单打包编译文件（手动打包）

Webpack 要打包路径 -o 存放路径

webpack .\src\main.js -o .\dist\bundle.js

1. 如何自动打包

安装 webpack-dev-server （需要依赖本地的 webpack ）

1. 创建 webpack.config.js

在node中运行 webpack的时候 工具会发现没有打包文件的进出口，没有会去检查根目录中 webpack.config.js 配置文件中去查找

具体配置信息有

首先利用 node将入口引入

const path = require(‘path’);

module.exports = { // 项目出口配置文件

entry: path.join(\_\_dirname, ‘./src/main.js’), // 入口文件

output: { // 项目出口文件

path: path.join(\_\_dirname, ‘./src’),

filename: ‘bundle.js’

}

}

便可以使用 webpack 打包，

再在pakeage.json 文件的 script 文件中添加

dev: “ webpack-dev-server --open(自动打开）--port 3000(端口）--contentBase src --hot

但是这样会导致打包编译的 bundle.js 文件变更目录，打包好的 bundle.js 文件会存在虚拟内存中

1. 所以可以安装 自动编译页面插件 html-webpack-plugin 它的功能： 在内存中，根据指定的模板页面，生成一份内存中的首页，同时把自动打包好的bundle注入到页面底部，

如果要配置插件，需要在导出的对象中，挂载一个plugins（这是所有的webpack插件的配置节点）

plugins: [ 所有的webpack 插件的配置节点

new htmlWebpackPlugin({

template: path.join(\_\_dirname, ‘./src/index.html’),// 指定模板文件路径

filename: ‘index.html’ // 设置生成的内存页面的名称

]

1. 接下来我们要引入CSS样式， 但是webpack无法自动打包，所以我们要引入第三方loader模块

cnpm i style-loader css-loader less-loader less -D

引入之后要在配置文件中配置第三方loader

module: {

rules: [

{test: /\.css$/, use: [‘style-loader’, ‘css-loader’]}

{test: /\.less$/, use:[‘style-loader’, ‘css-loader’, ‘less-loader’]}

]

}

1. 到达这一步我们想要引入 图片，但是webpack 默认无法处理URL 引入的图片或者文字库，所以需要引入插件

url-loader file-loader（后者是前者所依赖的插件）

webpack会自动将图片名字编码并且存储到虚拟内存中， 可以用传参的方式阻止

{test: /\.(jpg|jpeg|gif|png|bmp)$/, use: ‘url-loader?limit=...&name=[hash:..]-[name].[ext]

1. 如何引入字体文件呢？

首先需要安装字体文件的包，然后引入，我们这里用bootstrap做例子，注意： 如果通过路径的形式引用node\_modules的文件，可以省略掉node\_modules

{test:/\.(ttf|eot|svg|woff|woff2)$/, use: ‘url-loader’}

1. webpack只能识别部分ES6的语法，要使用则需要安装Babel 相关的 loader 功能，有两套功能。
   1. cnpm i babel-core babel-loader@7 babel-plugin-transform-runtime -D (需要指定版本)
   2. cnpm i babel-preset-env babel-preset-stage-0 -D

安装好后：

{test: /\.js$/, use: ‘babel-loader’, exclude:/node\_modules/}

配置babel的规则是，必须把node\_modules目录，通过 exclude排除，否则拖慢CPU并且将js 转码无法运行。

最后，在项目根目录中，新建一个 .babelrc的Babel 配置文件，这个文件属于json 格式，必须符合json 规范，不能写注释，必须双引号

{

“presets”: [“env”, “stage-0”],

“plugins”: [“transform-runtime”]

}